



Duplicate Detection On the Web

Finding Similar Looking Needles...
...In Very Large Haystacks

Sergei Vassilvitskii
Yahoo! Research, New York

November 15, 2007

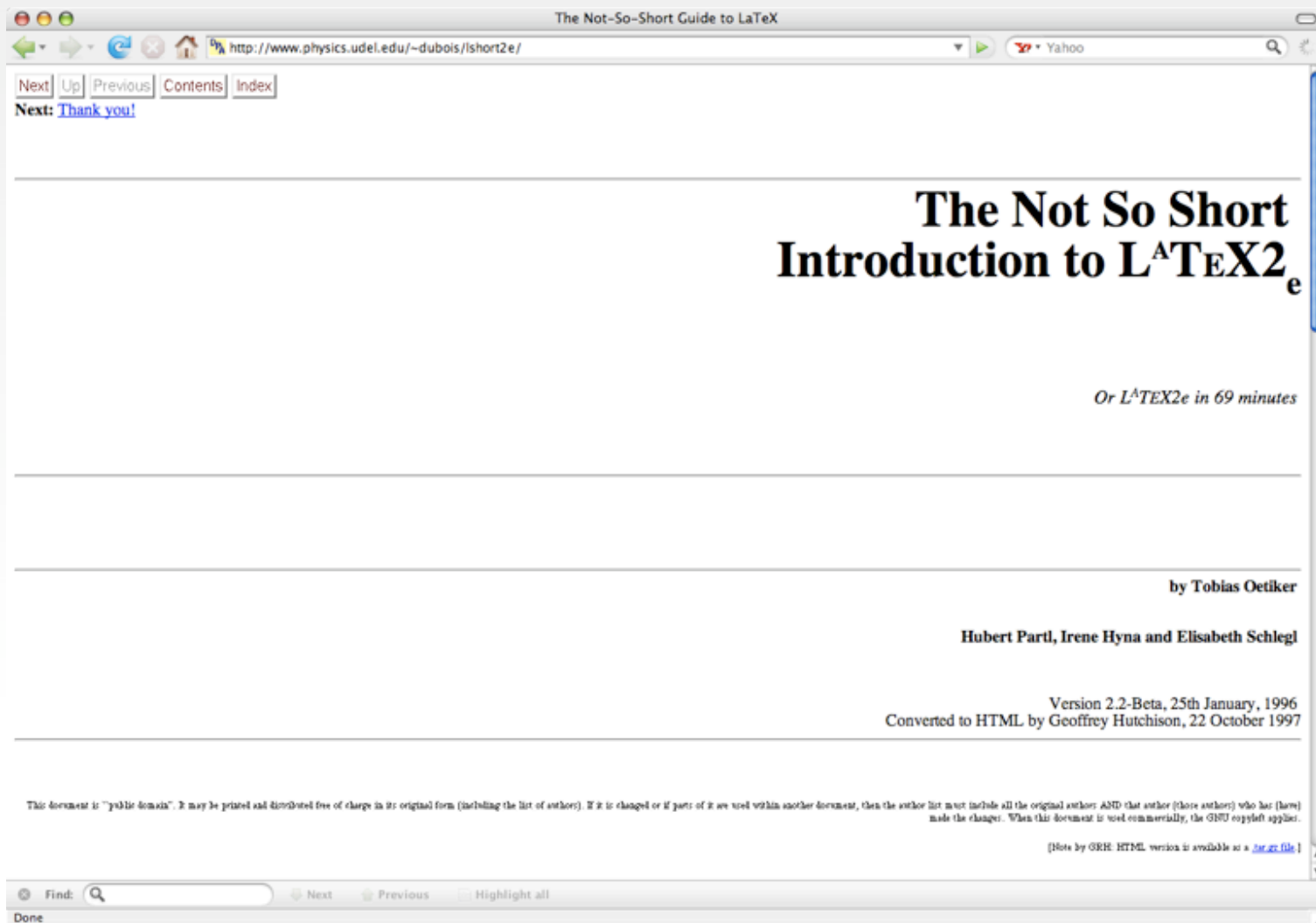
Duplicate Detection

- Why detect duplicates?
- Conserve resources
 - reduced index size - less memory, faster computations, etc.
- User Experience
 - Diversity in Search Results
 - 25-40% of the web is duplicate
 - Identical reviews with different boilerplate
 - mirrors - e.g. unix man pages
 - SPAM sites
 - etc etc

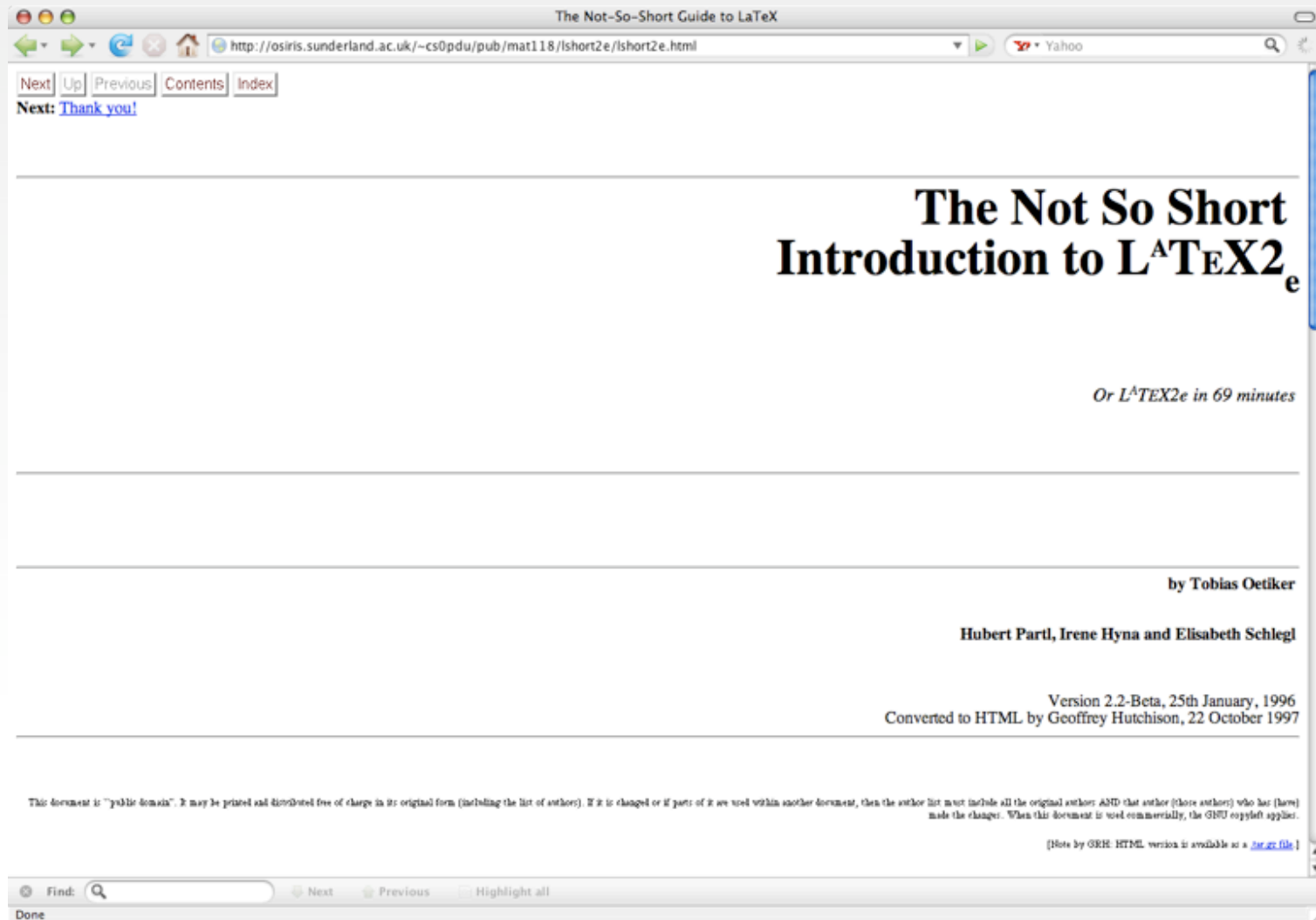
What's a Duplicate?

- Easy: exact duplicates

Exact Duplicates



Exact Duplicates



Exact Duplicates

The screenshot shows two overlapping browser windows. The top window is titled 'The Not-So-Short Guide to LaTeX' and has a URL bar showing 'http://www.physics.udel.edu/~dubois/lshort2e/'. The bottom window is also titled 'The Not-So-Short Guide to LaTeX' and has a URL bar showing 'http://osiris.sunderland.ac.uk/~cs0pdu/pub/mat118/lshort2e/lshort2e.html'. Two arrows originate from the title of the top window. One arrow points to a URL in the bottom window: 'http://osiris.sunderland.ac.uk/~cs0pdu/pub/mat118/lshort2e/lshort2e.html'. The other arrow points to another URL in the bottom window: 'http://www.physics.udel.edu/~dubois/lshort2e/'. The main content of the bottom window is the title page of 'The Not So Short Introduction to L^AT_EX₂_ε', which includes the authors 'Hubert Partl, Irene Hyna and Elisabeth Schlegl', the version 'Version 2.2-Beta, 25th January, 1996', and the conversion date 'Converted to HTML by Geoffrey Hutchison, 22 October 1997'. The page also contains a public domain notice and a search bar.

What's a Duplicate?

- Easy: exact duplicates
- Still easy: Different dates / signatures

Almost Duplicates

DBLP: Dragomir R. Radev

http://www.informatik.uni-trier.de/~ley/db/indices/a-tree/r/Radev:Dragomir_R.html

dblp.uni-trier.de

Dragomir R. Radev

List of publications from the DBLP Bibliography Server - FAQ

[Coauthor Index](#) - [Ask others](#): [ACM DL/Guide](#) - [CiteSeer](#) - [CSB](#) - [Google](#) - [MSN](#) - [Yahoo](#)

[Home Page](#)

2006

- 49 EE Siwei Shen, Dragomir R. Radev, Agam Patel, Günes Erkan: Adding Syntax to Dynamic Programming for Aligning Comparable Texts for the Generation of Paraphrases. *ACL 2006*
- 48 EE Dragomir R. Radev, Günes Erkan, Anthony Fader, Patrick Jordan, Siwei Shen, James P. Sweeney: LexNet: A Graphical Environment for Graph-Based NLP. *ACL 2006*
- 47 EE Rada Mihalcea, Dragomir R. Radev: Graph-based Algorithms for Natural Language Processing and Information Retrieval. *HLT-NAACL 2006*
- 46 EE Jahna Otterbacher, Dragomir R. Radev, Omer Kareem: News to go: hierarchical text summarization for mobile devices. *SIGIR 2006*: 589-596
- 45 EE Jahna Otterbacher, Dragomir R. Radev: Fact-focused novelty detection: a feasibility study. *SIGIR 2006*: 687-688

2005

- 44 EE Jahna Otterbacher, Günes Erkan, Dragomir R. Radev: Using Random Walks for Question-focused Sentence Retrieval. *HLT/EMNLP 2005*
- 43 EE Dragomir R. Radev, Omer Kareem, Jahna Otterbacher: Hierarchical text summarization for WAP-enabled mobile devices. *SIGIR 2005*: 679
- 42 EE Dragomir R. Radev, Jahna Otterbacher, Adam Winkel, Sasha Blair-Goldensohn: NewsInEssence: summarizing online news topics. *Commun. ACM 48(10)*: 95-98 (2005)
- 41 EE Wai Lam, Ki Chan, Dragomir R. Radev, Horacio Saggion, Simone Teufel: Context-based generic cross-lingual retrieval of documents and automated summaries. *JASIST 56(2)*: 129-139 (2005)
- 40 EE Dragomir R. Radev, Weiguo Fan, Hong Qi, Harris Wu, Amardeep Grewal: Probabilistic question answering on the Web. *JASIST 56(6)*: 571-583 (2005)

2004

- 39 EE Franz Josef Och, Daniel Gileade, Sanjeev Khudanpur, Anoop Sarkar, Kenji Yamada, Alexander Fraser, Shankar Kumar, Libin Shen, David Smith, Katherine Eng, Viren Jain, Zhen Jin, Dragomir R. Radev: A Smorgasbord of Features for Statistical Machine Translation. *HLT-NAACL 2004*: 161-168
- 38 EE Zhu Zhang, Dragomir R. Radev: Combining Labeled and Unlabeled Data for Learning Cross-Document Structural Relationships. *IJCNLP 2004*: 32-41
- 37 Harris Wu, Dragomir R. Radev, Weiguo Fan: Toward Answer-Focused Summarization Using Search Engines. *New Directions in Question Answering 2004*: 227-236
- 36 James Pustejovsky, Roser Sauri, José M. Castaño, Dragomir R. Radev, Robert J. Gaizauskas, Andrea Setzer, Beth Sundheim, Graham Katz: Representing Temporal and Event Knowledge for QA Systems. *New Directions in Question Answering 2004*: 99-112
- 35 EE Dragomir R. Radev, Hongyan Jing, Magorzata Sty, Daniel Tam: Centroid-based summarization of multiple documents. *Inf. Process. Manage.* 40(6): 919-938 (2004)
- 34 EE Günes Erkan, Dragomir R. Radev: LexRank: Graph-based Lexical Centrality as Salience in Text Summarization. *J. Artif. Intell. Res. (JAIR) 22*: 457-479 (2004)

2003

- 33 EE Dragomir R. Radev, Simone Teufel, Horacio Saggion, Wai Lam, John Blitzer, Hong Qi, Arda Çelebi, Danyu Liu, Elliott Drabek: Evaluation Challenges in Large-Scale Document Summarization. *ACL 2003*: 375-382

Find: Next Previous Highlight all

Done

Almost Duplicates

DBLP: Dragomir R. Radev

http://www.sigmod.org/dblp/db/indices/a-tree/r/Radev.Dragomir_R_.html

dblp.uni-trier.de

Dragomir R. Radev

List of publications from the DBLP Bibliography Server - FAQ

[Coauthor Index](#) - [Ask others](#): [ACM DL](#) - [ACM Guide](#) - [CiteSeer](#) - [CSB](#) - [Google](#)

[Home Page](#)

2006

- 49 [EE](#) Siwei Shen, Dragomir R. Radev, Agam Patel, Günes Erkan: Adding Syntax to Dynamic Programming for Aligning Comparable Texts for the Generation of Paraphrases. [ACL 2006](#)
- 48 [EE](#) Dragomir R. Radev, Günes Erkan, Anthony Fader, Patrick Jordan, Siwei Shen, James P. Sweeney: LexNet: A Graphical Environment for Graph-Based NLP. [ACL 2006](#)
- 47 [EE](#) Rada Mihalcea, Dragomir R. Radev: Graph-based Algorithms for Natural Language Processing and Information Retrieval. [HLT-NAACL 2006](#)
- 46 [EE](#) Jahna Otterbacher, Dragomir R. Radev, Omer Kareem: News to go: hierarchical text summarization for mobile devices. [SIGIR 2006](#): 589-596
- 45 [EE](#) Jahna Otterbacher, Dragomir R. Radev: Fact-focused novelty detection: a feasibility study. [SIGIR 2006](#): 687-688

2005

- 44 [EE](#) Jahna Otterbacher, Günes Erkan, Dragomir R. Radev: Using Random Walks for Question-focused Sentence Retrieval. [HLT/EMNLP 2005](#)
- 43 [EE](#) Dragomir R. Radev, Omer Kareem, Jahna Otterbacher: Hierarchical text summarization for WAP-enabled mobile devices. [SIGIR 2005](#): 679
- 42 [EE](#) Dragomir R. Radev, Jahna Otterbacher, Adam Winkel, Sasha Blair-Goldensohn: NewsInEssence: summarizing online news topics. [Commun. ACM 48\(10\)](#): 95-98 (2005)
- 41 [EE](#) Wai Lam, Ki Chan, Dragomir R. Radev, Horacio Saggion, Simone Teufel: Context-based generic cross-lingual retrieval of documents and automated summaries. [JASIST 56\(2\)](#): 129-139 (2005)
- 40 [EE](#) Dragomir R. Radev, Weiguo Fan, Hong Qi, Harris Wu, Amardeep Grewal: Probabilistic question answering on the Web. [JASIST 56\(6\)](#): 571-583 (2005)

2004

- 39 [EE](#) Franz Josef Och, Daniel Gileade, Sanjeev Khudanpur, Anoop Sarkar, Kenji Yamada, Alexander Fraser, Shankar Kumar, Libin Shen, David Smith, Katherine Eng, Viren Jain, Zhen Jin, Dragomir R. Radev: A Smorgasbord of Features for Statistical Machine Translation. [HLT-NAACL 2004](#): 161-168
- 38 [EE](#) Zhu Zhang, Dragomir R. Radev: Combining Labeled and Unlabeled Data for Learning Cross-Document Structural Relationships. [IJCNLP 2004](#): 32-41
- 37 [EE](#) Harris Wu, Dragomir R. Radev, Weiguo Fan: Toward Answer-Focused Summarization Using Search Engines. [New Directions in Question Answering 2004](#): 227-236
- 36 [EE](#) James Pustejovsky, Roser Sauri, José M. Castaño, Dragomir R. Radev, Robert J. Gaizauskas, Andrea Setzer, Beth Sundheim, Graham Katz: Representing Temporal and Event Knowledge for QA Systems. [New Directions in Question Answering 2004](#): 99-112
- 35 [EE](#) Dragomir R. Radev, Hongyan Jing, Magorzata Sty, Daniel Tam: Centroid-based summarization of multiple documents. [Inf. Process. Manage. 40\(6\)](#): 919-938 (2004)
- 34 [EE](#) Günes Erkan, Dragomir R. Radev: LexRank: Graph-based Lexical Centrality as Salience in Text Summarization. [J. Artif. Intell. Res. \(JAIR\) 22](#): 457-479 (2004)

2003

- 33 [EE](#) Dragomir R. Radev, Simone Teufel, Horacio Saggion, Wai Lam, John Blitzer, Hong Qi, Arda Çelebi, Danyu Liu, Elliott Drabek: Evaluation Challenges in Large-Scale Document Summarization. [ACL 2003](#): 375-382

Find: Next Previous Highlight all

Done

Almost Duplicates

DBLP: Dragomir R. Radev

http://www.sigmod.org/dblp/db/indices/a-tree/r/Radev:Dragomir_R=.html

80	Mark Sanderson	[26]
81	Anoop Sarkar	[39]
82	Roser Sauri	[29] [36]
83	Alan C. Schultz	[11]
84	Rich Schwartz	[26]
85	Sandip Sen	[11]
86	Andrea Setzer	[29] [36]
87	Libin Shen	[39]
88	Siwei Shen	[48] [49]
89	Amit Singhal	[26]
90	Alan F. Smeaton	[26]
91	David Smith	[39]
92	John R. Smith	[2] [7]
93	Magorzata Sty	[35]
94	Beth Sundheim	[36]
95	James P. Sweeney	[48]
96	Daniel Tam	[31] [35]
97	Simone Teufel	[24] [33] [41]
98	Howard R. Turtle	[26]
99	Evelyne Tzoukermann	[3] [4]
100	Ellen M. Voorhees	[26]
101	Ralph M. Weischedel	[26]
102	Adam Winkel	[23] [42]
103	Harris Wu	[22] [37] [40]
104	Jinxi Xu	[26]
105	Kenji Yamada	[39]
106	Cong Yu	[28]
107	Kazi A. Zaman	[2] [7]
108	ChengXiang Zhai (Chengxiang Zhai)	[26]
109	Zhu Zhang	[18] [19] [25] [32] [38]
110	Zhiping Zheng	[19]

Colors in the list of coauthors

DBLP: [Home | Search: Author, Title | Conferences | Journals]
 Michael Ley (ley@uni-trier.de) Thu Nov 1 04:38:51 2007

Find: Next Previous Highlight all

DBLP: [Home | Search: Author, Title | Conferences | Journals]
 Michael Ley (ley@uni-trier.de) Tue Nov 13 19:05:19 2007

Find: Next Previous Highlight all

Done

Almost Duplicates

DBLP: Dragomir R. Radev

http://www.sigmod.org/dblp/db/indices/a-tree/r/Radev.Dragomir_R=.html

80 Mark Sanderson [26]

81 Anoop Sarkar [39]

82 Roser Sauri [29] [36]

83 Alan C. Schultz [11]

84 Rich Schwartz [26]

85 Sandip Sen [11]

86 Andrea Setzer [29] [36]

87 Libin Shen [39]

88 Siwei Shen [48] [49]

89 Amit Singhal [26]

90 Alan E. Smeaton [26]

91 David Smith [94]

92 John R. Smith [29]

93 Magorzata Stry [31] [35]

94 Beth Sundheim [24] [33] [41]

95 James P. Sweeney [26]

96 Daniel Tam [35]

97 Simone Teufel [3] [4]

98 Ellen M. Voorhees [26]

99

100

101

102 Adam Winkel [42]

103 Harris Wu [37] [40]

104 Jinxi Xu [2] [2]

105 Kenji Yamada [39]

106 Cong Yu [28]

107 Kazi A. Zaman [2] [2]

108 ChengXiang Zhai (Chengxiang Zhai) [26]

109 Zhu Zhang [18] [19] [25] [32] [38]

110 Zhiping Zheng [19]

Colors in the list of coauthors

DBLP: [Home | Search: Author, Title | Conferences | Journals]
Michael Ley (ley@uni-trier.de) Thu Nov 1 04:38:51 2007

DBLP: [Home | Search: Author, Title | Conferences | Journals]
Michael Ley (ley@uni-trier.de) Tue Nov 13 19:05:19 2007

Find: Q Next Previous Highlight all

Done

What's a Duplicate?

- Easy: exact duplicates
- Still easy: Different dates / signatures
- Harder: Slight edits, modifications

is this a duplicate?



is this a duplicate?

The screenshot shows a web browser window with the address bar displaying `http://www.rootr.net/man/info/grep`. The page title is "man : grep". The browser's address bar also shows "Yahoo". The page content is a man page for the `grep` command. It includes sections for NAME, SYNOPSIS, DESCRIPTION, and a list of options. The page is styled with a blue header and a white body. The browser's status bar at the bottom shows "Done".

man : grep

Command: [man](#) [perldoc](#) [info](#) [search\(apropos\)](#) [Submit Query](#)

File: *manpages*, Node: grep, Up: (dir)

GREP(1) GREP(1)

OpenBSD Reference Manual

NAME

grep, egrep, fgrep, zgrep, zegrep, zfgrep - file pattern searcher

SYNOPSIS

grep [-abcEFGHhIiLlnoPqRSsUVvwXZ] [-A num] [-B num] [-C[num]]
[-e pattern] [-f file] [--binary-files=value] [--context[=num]]
[--line-buffered] [pattern] [file ...]

DESCRIPTION

The grep utility searches any given input files, selecting lines that match one or more patterns. By default, a pattern matches an input line if the regular expression (RE) in the pattern matches the input line without its trailing newline. An empty expression matches every line. Each input line that matches at least one of the patterns is written to the standard output.

grep is used for simple patterns and basic regular expressions (BREs); egrep can handle extended regular expressions (EREs). See [re_format\(7\)](#) for more information on regular expressions. fgrep is quicker than both grep and egrep, but can only handle fixed patterns (i.e. it does not interpret regular expressions). Patterns may consist of one or more lines, allowing any of the pattern lines to match a portion of the input.

zgrep, zegrep, and zfgrep act like grep, egrep, and fgrep, respectively, but accept input files compressed with the [compress\(1\)](#) or [gzip\(1\)](#) compression utilities.

The following options are available:

- A num Print num lines of trailing context after each match. See also the -B and -C options.
- a Treat all files as ASCII text. Normally grep will simply print "Binary file ... matches" if files contain binary characters. Use of this option forces grep to output lines matching the specified pattern.
- B num Print num lines of leading context before each match. See also the -A and -C options.
- b The offset in bytes of a matched pattern is displayed in front of the respective matched line.

Find: Next Previous Highlight all

Done

is this a duplicate?

The screenshot shows two browser windows side-by-side. The left window, titled 'Root R.NET', displays the 'man: grep' page. The right window, titled 'RTFM grep(1)', displays the 'MirOS Manual: grep(1)' page. Both pages show the same content: the name 'grep', its synopsis, and its description. The content is identical in both windows, suggesting a duplicate.

man: grep

Command: `grep`

File: `*manpages*`, Node: `grep`, ...

GREP(1) OpenB...

NAME

`grep`, `egrep`, `fgrep`, `zgrep`, ... `grep`, `egrep`, `fgrep`, `zgrep`, `zegrep`, `zfgrep` - file pattern searcher

SYNOPSIS

`grep [-abcEFGHhIiLlnoPqRSsUvwx] [-e pattern] [-f file] [--line-buffered] [pattern] [file ...]`

DESCRIPTION

The `grep` utility searches any given input files, selecting lines that match one or more patterns. By default, a pattern matches an input line if the regular expression (RE) in the pattern matches the input line without its trailing newline. An empty expression matches every line. Each input line that matches at least one of the patterns is written to the standard output.

`grep` is used for simple patterns and basic regular expressions (BREs); `egrep` can handle extended regular expressions (EREs). See [re_format\(7\)](#) for more information on regular expressions. `fgrep` is quicker than both `grep` and `egrep`, but can only handle fixed patterns (i.e. it does not interpret regular expressions). Patterns may consist of one or more lines, allowing any of the pattern lines to match a portion of the input.

`zgrep`, `zegrep`, and `zfgrep` act like `grep`, `egrep`, and `fgrep`, respectively, but accept input files compressed with the [compress\(1\)](#) or [gzip\(1\)](#) compression utilities.

The following options are available:

- `-A num` Print `num` lines of trailing context after each match. See also the `-B` and `-C` options.
- `-a` Treat all files as ASCII text. Normally `grep` will simply print "Binary file ... matches" if files contain binary characters. Use of this option forces `grep` to output lines matching the specified pattern.
- `-B num` Print `num` lines of leading context before each match. See also the `-A` and `-C` options.
- `-b` The offset in bytes of the respective match.

MirOS Manual: grep(1)

GREP(1) BSD Reference Manual GREP(1)

NAME

`grep`, `egrep`, `fgrep`, `zgrep`, `zegrep`, `zfgrep` - file pattern searcher

SYNOPSIS

`grep [-abcEFGHhIiLlnoPqRSsUvwx] [-A num] [-B num] [-C num] [-e pattern] [-f file] [--binary-files=value] [--context=num] [--line-buffered] [pattern] [file ...]`

DESCRIPTION

The `grep` utility searches any given input files, selecting lines that match one or more patterns. By default, a pattern matches an input line if the regular expression (RE) in the pattern matches the input line without its trailing newline. An empty expression matches every line. Each input line that matches at least one of the patterns is written to the standard output.

`grep` is used for simple patterns and basic regular expressions (BREs); `egrep` can handle extended regular expressions (EREs). See [re_format\(7\)](#) for more information on regular expressions. `fgrep` is quicker than both `grep` and `egrep`, but can only handle fixed patterns (i.e. it does not interpret regular expressions). Patterns may consist of one or more lines, allowing any of the pattern lines to match a portion of the input.

`zgrep`, `zegrep`, and `zfgrep` act like `grep`, `egrep`, and `fgrep`, respectively, but accept input files compressed with the [compress\(1\)](#) or [gzip\(1\)](#) compression utilities.

The following options are available:

- `-A num` Print `num` lines of trailing context after each match. See also the `-B` and `-C` options.
- `-a` Treat all files as ASCII text. Normally `grep` will simply print "Binary file ... matches" if files contain binary characters. Use of this option forces `grep` to output lines matching the specified pattern.
- `-B num` Print `num` lines of leading context before each match. See also the `-A` and `-C` options.

What's a Duplicate?

- Easy: exact duplicates
- Still easy: Different dates / signatures
- Harder: Slight edits, modifications
- Hardest: Different versions, updates, etc.

Similarity

- Tokens - trigrams in a document:

Once upon a midnight dreary, while I pondered weak and weary

Once upon a
upon a midnight
a midnight dreary
midnight dreary while

- Represent a document as a set:

{Once upon a, upon a midnight, a midnight dreary, ... }

- Similarity (A,B) = $\frac{|A \cap B|}{|A \cup B|}$

Similarity

- A: “Once upon a midnight dreary, while I pondered”
{ Once upon a, upon a midnight, a midnight dreary, midnight dreary while, dreary while I, while I pondered }
- B: “Once upon a time, while I pondered”
{ Once upon a, upon a time, a time while, time while I, while I pondered }

Similarity

- A: “Once upon a midnight dreary, while I pondered”
{ Once upon a, upon a midnight, a midnight dreary, midnight dreary while, dreary while I, while I pondered }
- B: “Once upon a time, while I pondered”
{ Once upon a, upon a time, a time while, time while I, while I pondered }
- Overlap: $|A \cap B| = 2$

Similarity

- A: “Once upon a midnight dreary, while I pondered”
{ Once upon a, upon a midnight, a midnight dreary, midnight dreary while, dreary while I, while I pondered }
- B: “Once upon a time, while I pondered”
{ Once upon a, upon a time, a time while, time while I, while I pondered }
- Overlap: $|A \cap B| = 2$
- Total: $|A \cup B| = 9$

Similarity

- A: “Once upon a midnight dreary, while I pondered”
{ Once upon a, upon a midnight, a midnight dreary, midnight dreary while, dreary while I, while I pondered }
- B: “Once upon a time, while I pondered”
{ Once upon a, upon a time, a time while, time while I, while I pondered }
- Overlap: $|A \cap B| = 2$
- Total: $|A \cup B| = 9$
- Similarity 0.22

Similarity

- Recall $\text{Sim}(A,B) = \frac{|A \cap B|}{|A \cup B|}$
 - Also known as the **Jaccard** similarity
- Is this a good similarity measure?
 - Yes: Simple to describe, easy to compute
 - No: Ignores repetition of trigrams:
 - e.g. “a rose is a rose” and “a rose is a rose is a rose” have 100% similarity.
 - Bad at detecting small but semantically important edits

Outline

- Motivation
- Algorithms
- Evaluation
- Open Problems



Algorithms

- Hashing:
 - Perfect for detecting exact duplicates. Doesn't work for near dupes.
- Edit distance
 - Perfect for detecting near dupes, does not scale.

Scale

- The size of the index is claimed to be 4B-16B pages
- Exact estimation is an active research topic
 - Also, bigger doesn't always mean better
- Safe to assume at least 1B pages in index.

Algorithms

- Hashing:
 - Perfect for detecting exact duplicates. Doesn't work for near dupes.
- Edit distance
 - Perfect for detecting near dupes, does not scale.
 - Compare every time.
- Shingling
 - We will store 48 bytes per page, detect near duplicates without examining all 1B pages.

Shingling Idea

- Computing Jaccard similarity is still expensive.
- Idea: summarize each document in a short **sketch**.
- Estimate the similarity based on the sketches.
- Algorithm due to Broder et al. (WWW '97), used in the Altavista search engine and all search engines since.

Algorithm

- Take a hashing function, H. Hash each shingle:

{ Once upon a, upon a midnight, a midnight dreary, midnight dreary while, dreary while I, while I pondered }

{ 357, 192, 755, 123, 987, 345 }

- Store the minimum hash: {192}

{ Once upon a, upon a time, a time while, time while I, while I pondered }

{ 357, 143, 986, 743, 345 }

- Store the minimum hash: {143}

Algorithm

- Repeat many times with different hash functions

	hash-1	hash-2	hash-3	hash-4
doc 1:	192	155	187	255
doc 2:	143	179	187	155

- Similarity - Percentage of times hashes agree
- $\text{SIM}(\text{doc 1}, \text{doc 2}) = 1/4$

Why Min-Hash

- Theorem:

$$\textit{Prob}[\text{Min-Hash}(A) = \text{Min-Hash}(B)] = \frac{|A \cap B|}{|A \cup B|}$$

- Therefore:
 - % of time the hashes agree $\sim \text{Sim}(A,B)$

Why Min-Hash

- Proof: Look at the elements:

{ Once upon a, upon a midnight, a midnight dreary, midnight dreary while, dreary while I, while I pondered, upon a time, a time while, time while I }

- One of these will have the minimum hash value.
- The two minima are the same if the corresponding trigram appears in both documents.
- Min-hashes are equal if either of { Once upon a, while I pondered } hashes to the minimum value.

Sketches

- So a sketch for a document is a collection of minimum hashes.
 - In practice, use 84 hash functions.
- sketch = { 192, 155, 187, 255, ..., 101 }
- Summarized each page in 672 bytes (84 8 byte values).

Sketches

- To compare two documents, look at the percentage of min-hashes that agree.
- Problem: Full Pairwise comparison
 - 10^9 pages by 10^9 pages by 10^2 hashes = 10^{20} operations.
- But we have many fast computers!
 - 10^9 operations / second * 10^4 machines would still require 10^7 seconds - roughly 4 months.

Super Shingles

- Problem: Doing all pairwise comparisons still too expensive.
- Solution: Since we care about only high similarity items, recurse:
- sketch = { 192, 155, 187, 255, 345, 171, 877, ... , 101 }
- Group into non overlapping super-shingles:

{192, 155, 187, 255}

{345, 171, 877, ...}

{..., 101}

- Hash each super-shingle: {1011, 6543, ..., 7327}
- Only compare documents that agree on super-shingles.

Super Shingles

	S-Shingle 1	S-Shingle 2	S-Shingle 3
Doc 1	1011	6543	7327
Doc 2	4523	5498	8754
Doc 3	5487	5498	8754

Super Shingles

	S-Shingle 1	S-Shingle 2	S-Shingle 3
Doc 1	1011	6543	7327
Doc 2	4523	5498	8754
Doc 3	5487	5498	8754

- Declare Doc2 and Doc3 to be 2-similar.
- In practice - store the above table sorted by different columns. Only compare against neighboring rows.

Super Shingles

- Store the super shingle table sorted by columns

	S-Shingle 1	S-Shingle 2	S-Shingle 3
Doc 1	1011	6543	7327
Doc 2	4523	5498	8754
Doc 3	5487	5498	8754
Doc 4	8766	1258	6255

	S-Shingle 1	S-Shingle 2	S-Shingle 3
Doc 4	8766	1258	6255
Doc 2	4523	5498	8754
Doc 3	5487	5498	8754
Doc 1	1011	6543	7327

- Only compare adjacent rows instead of all pairs

Summary

- Text = Once upon a midnight dreary, while I pondered ...
- Trigrams = { Once upon a, upon a midnight, a midnight dreary, midnight dreary while, dreary while I, while I pondered, ... }
- Hashes(1) = { 357, 192, 755, 123, 987, 345, ... }
- Min Hash(1) = { 192 }
- Hashes(2) = { 132, 345, 487, 564, 778, 120, ... }
- Min Hash(2) = { 120 }
- ..
- Min Hash(84) = { 101 }
- Sketch = { 192, 120, ..., 101 }
- Super Shingles = {1011, 6543, 7327, 5422, 8764, 2344}
- Similar if exact overlap on 2 or more super shingles.

Outline

- Motivation
- Algorithms
- Evaluation
- Open Problems



Evaluation

- Due to Henzinger, SIGIR '06
- Start with 1.6B webpages - 46M hosts, on average 35 pages per host.
- Remove exact duplicates (around 25%)
- Impossible to check recall of the algorithms (Why?)
- To check precision: sample roughly 2000 pairs returned as duplicates and evaluate

Evaluation

- Pairs are decided to be near duplicate if:
 - Text differs by timestamp, visitor count, etc.
 - Difference is invisible to the visitor
 - Entry pages to the same site
- Not near duplicate if:
 - Main items are different (e.g. shopping page for two different items, but with identical boilerplate text)

Evaluation

- Undecided:
 - Prefilled forms with different values
 - A different 'minor' item - e.g. small text box
 - Pairs that could not be evaluated (e.g. english speaking evaluator looking at two Korean pages)

Results

Precision on the 'duplicate' set.

	Pairs	Correct	Incorrect	Undecided
All	1910	0.38	0.53	0.09
2-sim	1032	0.24	0.68	0.08
3-sim	389	0.42	0.48	0.1
4-sim	240	0.55	0.36	0.09
5-sim	143	0.71	0.25	0.06
6-sim	106	0.85	0.05	0.1

Outline

- Motivation
- Algorithms
- Evaluation
- Open Problems



Open Problems

- The devil is always in the details:
 - Obtaining text from raw HTML is not as easy as it sounds
 - What to do with IMG ALT text? Targets of links? etc.
 - Large boilerplate text with few seemingly minor differences
- New Challenges
 - Dynamic content?
 - Flash, Ajax, and other not easily indexable content

References

- Broder, Glassman, Manasse, Zweif. [Syntactic clustering of the web](#). WWW '97.
- Fetterly, Manasse, Majork. [Detecting phrase level duplication on the World Wide Web](#). SIGIR '05.
- Henzinger. [Finding near duplicate Web pages: a large scale evaluation of algorithms](#). SIGIR '06.





Thank You!